

DLCV2

Hamann, Clarissa

November 2019

1 Baseline Model

1.1 Preprocessing

For the pretrained Resnet images have to be loaded in to a range of $[0, 1]$ and then normalized using $\text{mean} = [0.485, 0.456, 0.406]$ and $\text{std} = [0.229, 0.224, 0.225]$.

For the baseline model no other augmentation is used.

1.2 Figures

At first the layers without resnet got trained. Afterwards the whole network with the pretrained resnet got trained. This increased the accuracy.

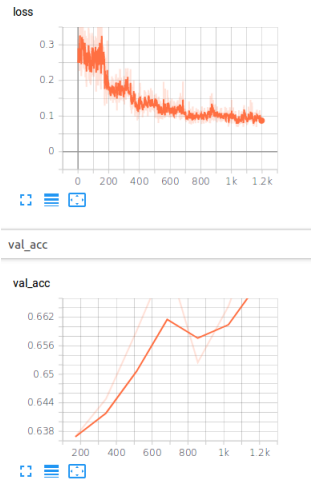


Figure 1: Training loss and IoU Score (acc), first Training

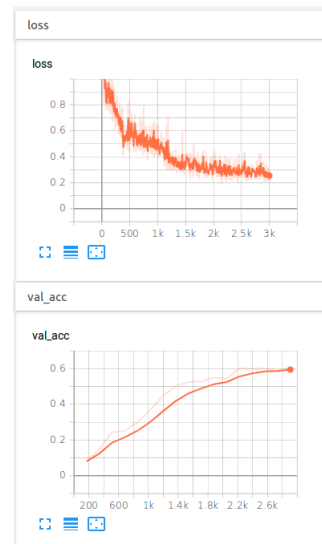


Figure 2: Training loss and IoU Score (acc), second Training

1.3 semantic segmentation result



Figure 3: classified as class 1

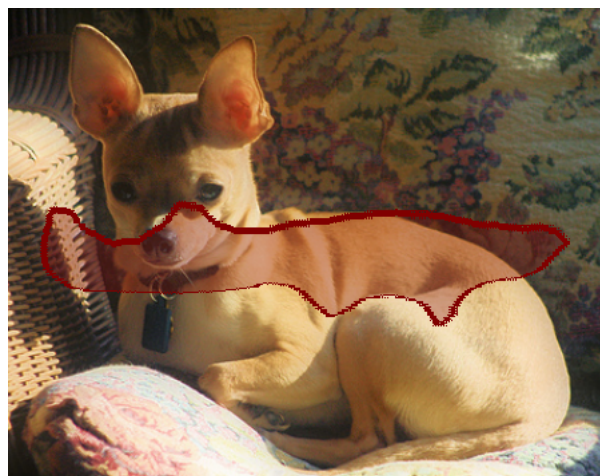


Figure 4: classified as class 2

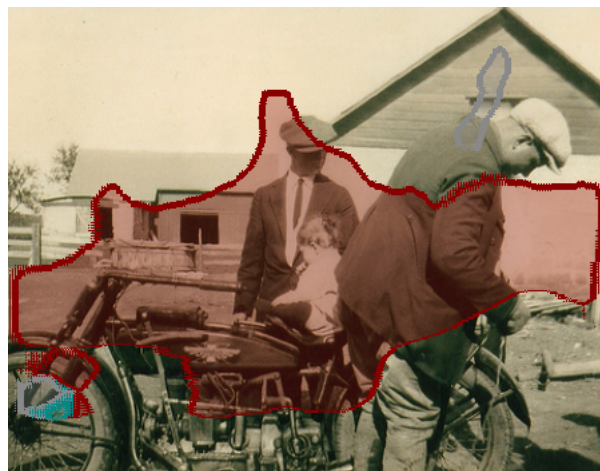


Figure 5: classified as class 2,3,8



Figure 6: classified as class 4



Figure 7: classified as class 5, 6

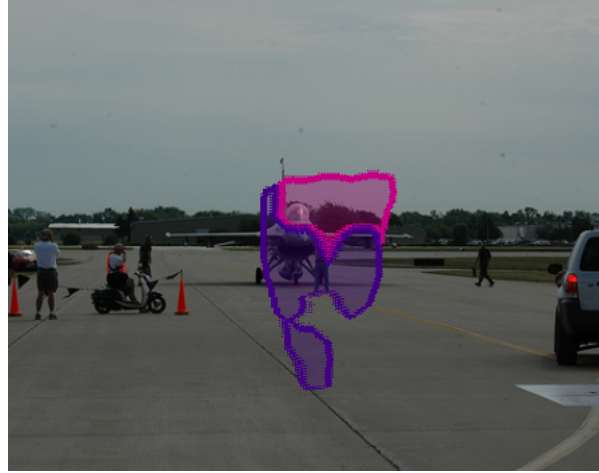


Figure 8: classified as class 5, 6



Figure 9: classified as class 7

1.4 mIoU score

The model's best class is 0: background, the worst one is class 4:cat. Background is the best, because it appears in every image and due to this it has the most data. To classify a cat is difficult for the model, because it has to classify an animal and the kind of animal. The the model has problems with distinguishing the animals cats and dogs.

```

-----
class #0 : 0.90244
class #1 : 0.74702
class #2 : 0.70119
class #3 : 0.71168
class #4 : 0.38764
class #5 : 0.58934
class #6 : 0.66698
class #7 : 0.73553
class #8 : 0.69780

mean_iou: 0.682180

Testing Accuracy: 0.6821796014923149

```

Figure 10: IoU

2 Improved model

2.1 Model architecture

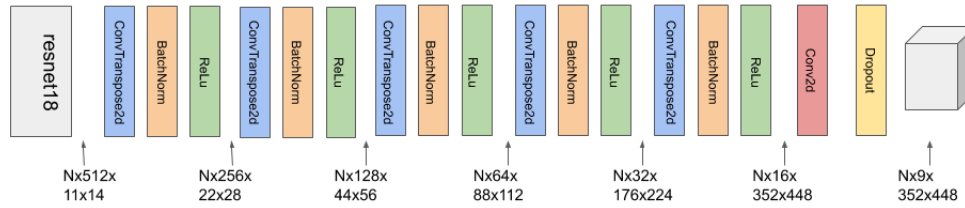


Figure 11: Model architecture

Transpose Convolution layer (kernel size=4, stride=2, padding=1, bias=False)
Convolution layer (kernel size=1, stride=1, padding=0, bias=True)

2.2 Why is the model better

For this improved model augmentation, like rotation of the data, and batch normalization is used. Rotation of images leads to more divers data and the

model not just learns the obvious shapes of the classes and figures (e.g. figure 3 put shape of person and 4 shape of plane). Here a rotation of 30 degree is used. In addition to the layers from the baseline model some batch normalization got added before the ReLU layers. It makes it possible to use a smaller learning rate and this leads to a faster training. Batch normalization allows each layer of a network to learn by itself a little bit more independently of other layers. Batch normalization also makes sure that no activation get really high or low. An other advantage is that it reduces overfitting, because it adds some noise to each hidden layers activation.

The beginning we started an end-to-end training with different learning rates. It could not reach the target accuracy. It did not worked well enough as you can see in following figures.

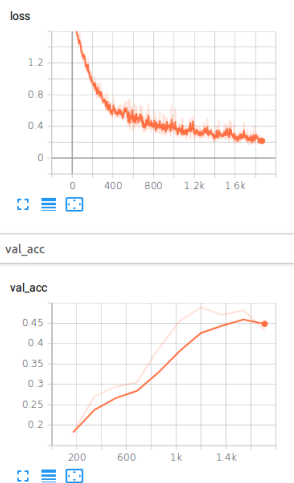


Figure 12: end-to-end

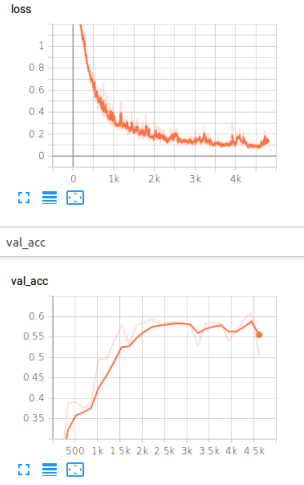


Figure 13: end-to-end

Then the model got trained with freezed resnet weights and a higher learning rate (part 1) and afterwards the training got started again with the complete model (no freezed weights), a dropout layer in the end and a smaller learning rate (part 2). The dropout layer leads to more generalization. If some nodes are already got trained very well, they get dropped this node out, that weak nodes (weak classes in our case class 4) get more trained for a while. Maybe one improvement could be to add this layer already in the beginning of the training.

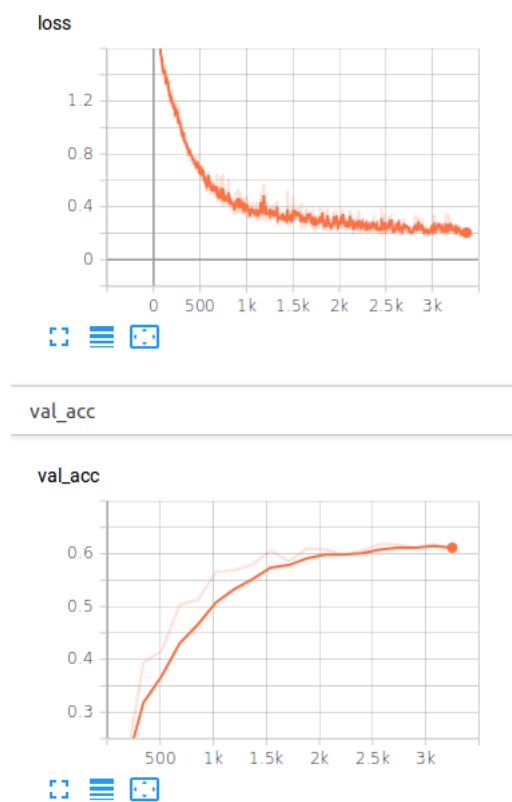


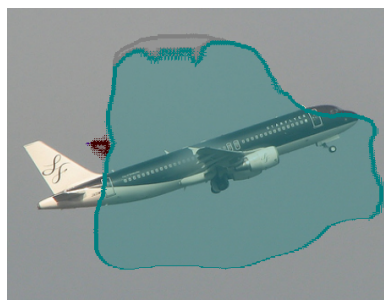
Figure 14: part1; freezed resnet weights

2.3 mIoU score + results

```
class #0 : 0.90166
class #1 : 0.73932
class #2 : 0.67720
class #3 : 0.70505
class #4 : 0.43750
class #5 : 0.61821
class #6 : 0.68852
class #7 : 0.74057
class #8 : 0.69687

mean_iou: 0.689432
```

Figure 15: IoU



(a) prediction



(b) ground truth

Figure 16: plane



(a) prediction



(b) ground truth

Figure 17: plane



(a) prediction



(b) ground truth

Figure 18: plane

3 Bonus

3.1

$$\begin{aligned} G(x) &= \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2}{2\sigma^2}} \cdot \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{y^2}{2\sigma^2}} \\ &= \frac{1}{2\pi\sigma^2} \cdot \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2}{2\sigma^2} - \frac{y^2}{2\sigma^2}} \\ &= \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2+y^2}{2\sigma^2}} \\ &= G(x, y) \end{aligned} \tag{1}$$

3.2 2D Gaussian Filter

The Gaussian filter is used to blur the image. It is used to reduce noise and blur details. The bigger sigma the bigger is the blurring effect.



Figure 19: Gauss

3.3

$$K_x = \begin{pmatrix} -0.5 \\ 0 \\ 0.5 \end{pmatrix} \tag{2}$$

$$K_y = \begin{pmatrix} -0.5 & 0 & 0.5 \end{pmatrix} \quad (3)$$

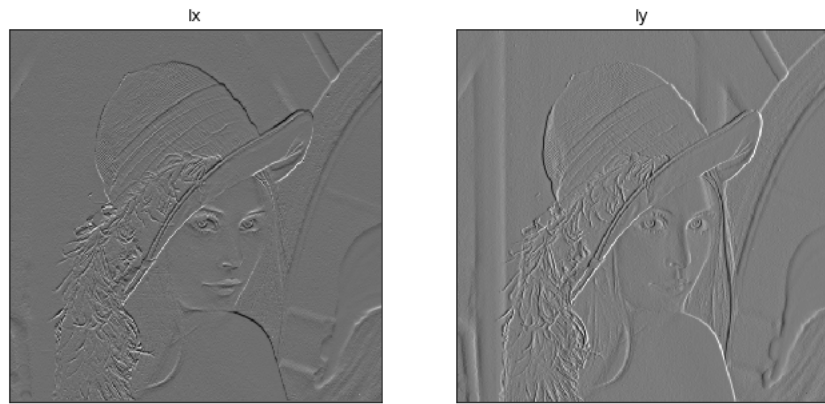


Figure 20: Edge detection

3.4

The image with the Gaussian filter shows more blurred edges than the images without.

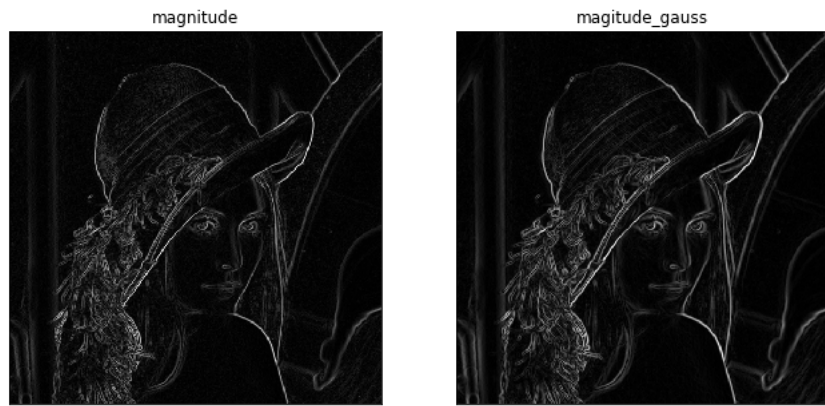


Figure 21: Magnitude

References

In cooperation with Chiara Pullem (A08323104). For this work the documentary of pytorch, stackoverflow and Towards Data Science are used.